Refine Search

Search Results -

Terms	Documents
L11 same suspend$	5

Database:
US Pre-Grant Publication Full-Text Database
US Patents Full-Text Database
US OCR Full-Text Database
EPO Abstracts Database
JPO Abstracts Database
Derwent World Patents Index
IBM Technical Disclosure Bulletins

Search: L12

Refine Search

Recall Text | Clear | Interrupt

Search History

DATE: Monday, June 14, 2004 Printable Copy Create Case

Set Name side by side	Query	Hit Count	Set Name result set
DB=USPT; PLUR=YES; OP=OR			
L12	L11 same suspend$	5	L12
L11	L10 same host	322	L11
L10	test$ near3 processor	5569	L10
L9	L6 same host	20	L9
L8	L7 and l6	20	L8
L7	host adj1 processor	8157	L7
L6	L5 same test$	74	L6
L5	suspend$ near3 processor	1102	L5
L4	L3 and l2	18	L4
L3	motherboard same interface	2308	L3
L2	L1 same suspend$	220	L2
L1	test$ same processor	23549	L1

END OF SEARCH HISTORY

L8: Entry 1 of 20 File: USPT Nov 4, 2003

DOCUMENT-IDENTIFIER: US 6643803 B1
TITLE: Emulation suspend mode with instruction jamming

Brief Summary Text (11):
A digital system is provided having a processor, wherein the processor is operable to execute a sequence of instructions obtained from an instruction bus connected to an instruction memory circuit, wherein the processor comprises. The processor has system resources including registers and at least one memory circuit. The processor has test port circuitry for receiving test commands from a remote test host. Emulation circuitry for debug events is connected to the test port circuitry, operable to cause the processor to enter a debug suspend state in response to a debug event and to leave the debug suspend state to resume execution of the sequence of instruction in response to a command received by the test port circuitry, wherein execution of the sequence of instructions ceases while the processor is in the debug suspend state. The emulation circuitry is operable to access the system resources in response to an access command received by the test port circuitry. while the processor is executing the sequence of instructions such that execution of the sequence of instructions is not delayed by the access of the system resources.

Detailed Description Text (12):
The following definitions will help the reader to understand the information in the rest of this application: Background code. The body of code that can be halted during debugging because it is not time-critical. Foreground code. The code of time-critical interrupt service routines, which are executed even when background code is halted. Debug-halt state. The state in which the device does not execute background code. Time-critical interrupt. An interrupt that must be serviced even when background code is halted. For example, a time-critical interrupt might service a motor controller or a high-speed timer. Debug event. An action, such as the decoding of a software breakpoint instruction, the occurrence of an analysis breakpoint/watchpoint, or a request from a host processor that can result in special debug behavior, such as halting the device or pulsing one of the signals EMU0 or EMU1. Break event. A debug event that causes the device to enter the debug-halt state.

Detailed Description Text (116):
FIG. 18 is a block diagram illustrating an alternative embodiment of a digital system with DSP 100, according to aspects of the present invention. Digital system 1500 includes processor 100, as described earlier, and a second processor referred to as host processor 1510.

Detailed Description Text (117):
DSP core 100 performs digital signal processing related tasks, while host processor 1510 performs other application tasks. DSP 100 is connected to an internal program memory. circuit 812 and to a dual ported communication memory circuit 1502 via bus 1530. Bridge 1503 is also connected to bus 1530 and provides access to peripherals 1520 and 1521 via bus 1531. Access is also provided via bus 1531 to dedicated hardware 1522, which includes various devices and circuits, such as timers, power controls, debugging and emulation circuitry, and such. Interrupt request signals 1540 provide interrupt requests from devices 1520-1522 to DSP 100.

Detailed Description Text (119):
Host processor 1510 is connected to host processor interface circuit (HPI) 1511 via bus 1533. HPI 1511 provides buffering and timing control to allow host processor 1510 to access communication memory circuit 1502 via bus 1532. In this manner, host processor 1510 can store and access data values in communication memory 1502 that can also be stored and accessed by DSP 100. Bus 1532 is separate from bus 1530 and communication memory 1502 is arranged such that host processor 1510 can access data values in dual ported memory circuit 1502 in a manner that does not impinge on the operation of memory circuit 812. Interrupt request signals 1541 provide interrupt requests from host processor 1510 to DSP 100.

CLAIMS:

1. A digital system comprising a processor, wherein the processor is operable to execute a sequence of instructions obtained from an instruction bus connected to an instruction memory circuit, wherein the processor comprises: system resources including registers and at least one memory circuit; an instruction pipeline operable to decode the sequence of instructions, access operands from system resources, and store results in the system resources; test port circuitry for receiving test commands from a remote test host; emulation circuitry for debug events connected to the test port circuitry, operable to cause the processor to enter a debug suspend state in response to a debut event and to leave the debug suspend state to resume execution of the sequence of instruction in response to a command received by the test port circuitry, wherein execution of the sequence of instructions ceases while the processor is in the debug suspend state; and further comprising detection circuitry operable to detect a bubble in the instruction pipeline during which no system resource is being accessed in response to executing the sequence of instructions; and jamming circuitry connected to the instruction pipeline operable to jam an access for a system resource in response to an access command received by the test port circuitry into the bubble detected by the detection circuitry, whereby the access of the system resources in response to the access command is performed without delaying the instruction pipeline when the processor is not in the debug suspend state.